(5 points) What is the language generated by the following grammar (where a and b are terminal symbols and S is the root symbol):

3/5

$$S ::= aSa \mid aBa$$
$$B ::= bB \mid b$$

Palandrom
The order of reading from left to right is same as right to left. The center is must be 'b'.

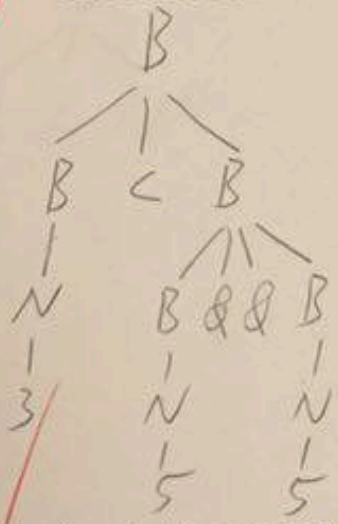
But not bbb or cbabab

(15 points) Suppose you have the following grammar:

$$B ::= B < B \mid B \,\&\&\, B \mid N$$

where N stands for all integer numbers.

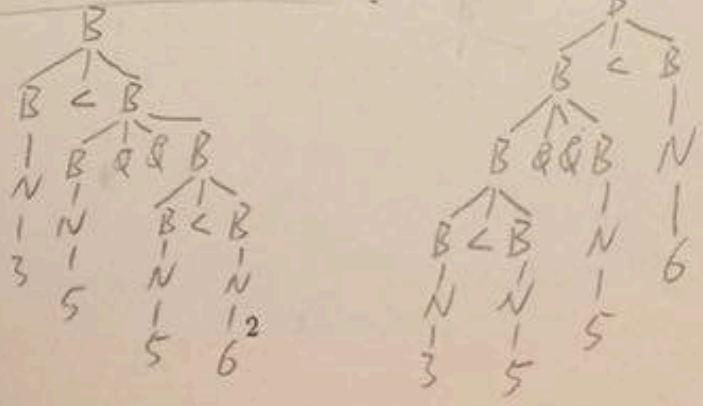
– Give a parse tree for the expression $3 < 5 \,\&\&\, 5$.



– Show that the grammar is ambiguous.

B < B && B < B

3 < 5 && 5 < 6



13

– Rewrite the grammar

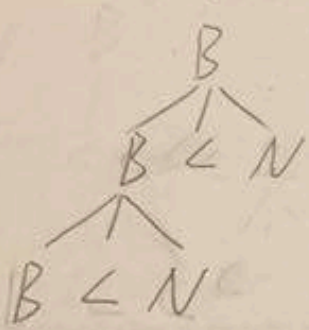
$$B ::= B < B \mid B \,\&\&\, B \mid N$$

so that both < and && are left associative, and < has higher precedence than &&.

4/5

$B ::= B < N \mid B \&\& N \mid N$

$3 < 5 \&\& 5 < 6$



4

(2 points) Give an example of a higher-order function in ML.

```
val x=4;
fun f(y) = x*y;
fun g(h) = let val x=7 in h(3)+x;
g(f);
```

(4 points) Give an example of a curried function in ML, and its un-curried form.

curry:

```
('a*'b)->'c)->('a->('b->'c))
```

uncurry:

```
('a->('b->'c))->(('a*'b)->'c)
```


6

(10 points) Consider the following ML definitions:

```
fun fact n = if n = 0 then 1 else n * fact (n-1);
fun new_if (a,b,c) = if a then b else c
```

Consider the new function `new_fact` using `new_if` instead of the if expression:

```
fun new_fact n = new_if (n=0, 1, n * new_fact (n-1))
```

1 Explain why `new_fact` does not compute the factorial. Think about the result of `new_fact 3`.

0

new_fact 3 = 3*2*1*f(0)

It times 1 twice.

2/5

2 Change the definitions of `new_if` and `new_fact` to make `new_fact` work correctly (the change allowed only deals with the formal and actual parameters of `new_if`).

fun new_fact n = new_if(n=1, 1, n*new_fact(n-1))

But now new_fact 0 would be wrong (if it didn't loop!)

2

Given the list data type :

```
datatype 'a list = Empty | Cons of 'a * 'a list
```

(8 points) Write a function map in ML that takes a function as an argument and a list and returns a new list by mapping the values of the list to the new values, using the function passed in as a parameter. For example,

```
map ((fn x => x + 1), (Cons (1, Cons (2, Empty))))
```

should evaluate to `Cons(2, Cons (3, Empty))`

6/8

```
fun map(fn,nil) = []
| map(fn,h::t) = fn h :: map(fn,t)
```

(5 points) Write an ML function

```
add_list : int list -> int
```

that adds the numbers up in a list.
For example, `add_list (Cons (2, Cons (3, Cons(4, Empty))))` returns 9.

```
fun add_list nil = 0
| add_list (h::t) = h + add_list(t);
```


11

(5 points) Using *only* the function defined below:

```
fun reduce F nil     y = y
  | reduce F (x::xs) y = F (x, (reduce F xs y))
val reduce = fn : ('a * 'b -> 'b) -> 'a list -> 'b -> 'b
```

write a function `sum-list` which given a list of natural numbers (i.e. positive integers) returns the sum of all its elements. It should returns 0 if the list is empty. For example,

```
sum-list [1,2,3,4] = 10
```

Your `sum-list` function should only call function `reduce` with the appropriate parameters. (This function `reduce` is similar to the function `reduce` you wrote in the last assignment. The difference being that in here we have an additional parameter `y` of type `'b` and we deal with lists instead of trees.)

```
fun sum-list  list = reduce .....................................
```

```
fun sum_list nil = 0
| sum_list [h::t] = h + sum_list [t];
```

(5 points) Given the following definitions:

```
datatype Seq = Nil | Cons of int * (unit -> Seq)

fun head (Cons (x, _)) = x;

fun tail (Cons (_, xs)) = xs ();
```

Write a function `elemN` which returns the n-th element of a sequence. For example,

```
elemN 2 (Cons(3, fn ()=> Cons (4, fn ()=> Cons(5, fn ()=>Nil))))
```

will return 5. You can return 99 if the index is out of bound,

```
elemN 3 (Cons(3, fn ()=> Cons (4, fn ()=> Nil)))
```

will return 99.

fun elemN n = head

(5 points) Write a ML datatype PERSON that defines a person by its first name, last name, age and date of birth.

2/5

datatype aPERSON = Firstname of string | Lastname of string

| Date of int * int * int | Age of int

(5 points) Write a ML datatype `tree` that defines a polymorphic binary tree. Assume that the values are contained in both the leaves and the interior nodes of the tree.

4/5

datatype 'a tree = Leaves of 'a

| Node of 'a tree * 'a tree

* 1/2

10